Journal of Applied Research and Technology

www.jart.icat.unam.mx

ICAT Instituto de Ciencias Aplicadas y Tecnología

*Journal of Applied Research and Technology* 24 (2026) 13-25

Original

# Discrete-Time Integral LQR and $\mathscr{H}_\infty$ Control of a Two-Wheeled Self-Balancing Robot

Filipe H. Varaschim[1] • Ricardo Breganon*[1]
Uiliam Nelson L. T. Alves[1]

*[1]IFPR – Federal Institute of Paraná, Av. Dr. Tito, 801, Jacarezinho – PR, Brazil.*

**Received: 05 21 2024; Accepted: 02 18 2025**
**Available: 12 31 2025**

**Abstract:** Inverted pendulum systems are widely used in both education and research within control theory. There are different types of inverted pendulums, such as the Furuta pendulum, cart-pole system, reaction wheel pendulum, etc. This study investigates the dynamics of a Two-Wheeled Self-Balancing Robot, a complex electromechanical system characterized by inherently nonlinear dynamics, unstable equilibrium points, and underactuation (i.e., more degrees of freedom than control inputs), inspired by inverted pendulum systems. The primary objective pursued in this investigation is to guide the robot along a predefined trajectory while maintaining its vertical orientation. To achieve this, we designed and implemented two controllers with an integrator in the feedback loop for a Two-Wheeled Self-Balancing Robot prototype. One controller considers the Linear Quadratic Regulator (LQR) technique, and the second is based on the $\mathscr{H}_\infty$ approach. Due to the use of a microcontroller platform, we designed the controller in discrete time to implement it. Ramp-like and sinusoidal references were used to set the robot's displacement targets, aiming to track these references while maintaining its vertical upright position. Simulations and practical tests were conducted using the Simulink® software within the Matlab® environment to evaluate the effectiveness of the proposed control strategies. The performances of the closed-loop system using each controller are compared. The results demonstrate the capability of the designed controllers to achieve the predefined objective, even in the presence

*Corresponding author.
*E-mail address:* ricardo.breganon@ifpr.edu.br (R. Breganon).
Peer Review under the responsibility of Universidad Nacional Autónoma de México.

of noise and inherent imperfections within the prototype. The $\mathscr{H}_\infty$ controller showed equal or better performance than the LQR controller for different indices in the experimental tests.

*Keywords:* **LQR Control. $\mathscr{H}_\infty$ Control. Discrete Control. Integral Control. Inverted Pendulum. Two-Wheeled Self-Balancing Robot.**

## 1. Introduction

Inverted pendulum systems are widely used in both education and research within control theory, serving as examples of nonlinear and underactuated electromechanical systems with an unstable equilibrium point (Graichen et al., 2007). Moreover, inverted pendulums have numerous applications across industries, including military, aeronautics, and aerospace, serving as missile and rocket launchers, robots, etc. (Hazem et al., 2020; Pedroso & Modesto, 2017).

The inverted mobile pendulum has been utilized, for instance, as a mode of human transportation, which could present itself as an alternative for automatic transportation in urban areas, including its utilization by individuals with physical disabilities (Tirmant et al., 2002). Other examples of works employing this type of system include Sun & Gan (2010), who utilize LQR combined with PID for balancing an inverted pendulum robot; Hehn & Andrea (2011), who use a drone for controlling an inverted pendulum; Cho (2021), who employs inverted pendulum concepts in missile orientation for stationary target interception, and Kim et al. (2005), who work on the modeling and LQR control of an inverted pendulum robot.

Optimal control, a modern control theory, is widely utilized across various engineering domains, including aerospace, automotive, robotics, and other applications (Park et al., 2019). The Linear Quadratic Regulator (LQR) is an optimal control method known for its robust nature. However, LQR may exhibit steady-state error (Dhewa et al., 2022). To address this issue, an integrator can be incorporated to reduce the steady-state error by adding a term based on the integration of the error (Jeong et al., 2013). This form of control, known as servo system, finds extensive use in the literature and diverse applications, as evidenced by studies such as Yamanaka et al. (2022), who worked with the control of an aeropendulum prototype, and Niro et al. (2017), who dealt with a ball-beam system.

The $\mathscr{H}_\infty$ control allows for mitigating the effect of existing disturbances in the system dynamics (Boyd et al., 1994) and has been used in different systems such as active suspension system (de Oliveira et al., 2018), Furuta pendulum (Alves et al., 2022b), Aeropendulum (Breganon et al., 2023), etc.

Discrete-time systems are widely used in engineering and other fields for modeling, analyzing, and implementing control strategies. They provide a practical and effective approach to dealing with real systems and their sampling and processing limitations. The discrete-time domain enables the analysis of signals over short sampling intervals (Moon et al., 2003).

This study focuses on controlling a Two-Wheeled Self-Balancing Robot, a system inspired by inverted pendulum systems. The objective is to make the robot maintain vertical balance using gyroscopes, accelerometers (MPU 6050), and encoders, while tracking displacement trajectories. The prototype employs an Arduino Mega and a Minseg Shield for data acquisition. Sensor calibration, motor synchronization, and the development of discrete-time controllers with an integrator in the feedback loop are addressed in this paper. The first designed controller considers the LQR technique, and the second one is based on the $\mathscr{H}_\infty$ theory. In contrast to previously published works such as Varaschim et al. (2023) and Valadão (2022), where the primary objective is to maintain the robot's balance only, this study aims to enable the robot to follow a desired displacement trajectory while maintaining balance. The designed controllers are tested in simulations and real experiments in a prototype, where the performance of the closed-loop system is measured by different indices and compared.

The paper follows presenting the Two-Wheeled Self-Balancing Robot prototype, its operating principle, main parameters, component description, and modeling in Section 2. Section 3 designs discrete-time LQR and $\mathscr{H}_\infty$ controllers with an integrator in the feedback control loop aiming to maintain the robot's balance while it follows a specific displacement trajectory. This section also demonstrates how experiments with the prototype were conducted. Section 4 presents and analyzes the results obtained from simulations and experiments. Section 5 concludes the paper.

## 2. Two-Wheeled Self-Balancing Robot

### 2.1 Prototype

This paper uses the Two-Wheeled Self-Balancing Robot from Minseg, as shown in Figure 1 (Minseg, 2021). The prototype is based on the Arduino platform and features sensors and actuators such as the MPU 6050 and two N20 motors with encoders. The robot is equipped with the Minseg Shield, which facilitates the connection and data acquisition of the sensors and motor drives.

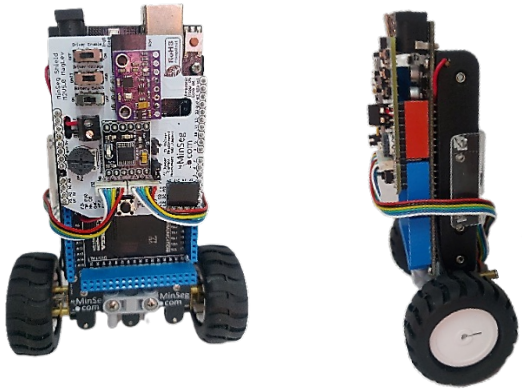

Figure 1. Minseg Two-Wheeled Self-Balancing Robot M2V5 available at the Industrial Automation and Control Laboratory of the Federal Institute of Paraná, Jacarezinho.

### 2.2 Prototype Parameters' Nomenclature

- $g$ – Gravity acceleration
- $P$ – Center of mass of the robot
- $\alpha$ – Robot tilt angle
- $h$ – Total height of the robot
- $L$ – Distance from the wheel axis to the point $P$
- $m_p$ – Mass of the robot
- $m_w$ – Wheel's mass
- $I_p$ – Moment of inertia of the robot at point $P$
- $I_w$ – Moment of inertia at the center of mass of the wheel
- $r_w$ – Wheel's radius
- $x$ – Linear displacement of the robot
- $E_c$ – Encoder conversion gain
- $pm$ – Pulses per revolution of the encoder
- $R$ – Electrical resistance of the motor's armature
- $V$ – Motor voltage
- $K_t$ – Motor's torque constant
- $K_b$ – Electromotive force constant
- $i$ – Current in the motor's armature
- $\theta$ – Angle of the robot wheel with respect to its center
- $\omega$ – Angular velocity of the motor's shaft

### 2.3 Modeling

We utilized the model published by Howard & Bushnell (2015). The modeling approach starts with the electrical subsystem and then connects it with the mechanical dynamics. The resulting equations from the Howard & Bushnell (2015) model are:

$$\left(\mathrm{I_p} + \mathrm{L^2 m_p}\right)\ddot{\alpha} + \mathrm{L m_p}\ddot{\mathrm{x}} = -\frac{\mathrm{K_t}}{\mathrm{R}}\mathrm{V} + \mathrm{g L m_p}\alpha + \frac{\mathrm{K_b K_t}}{\mathrm{R r_w}}\dot{\mathrm{x}} - \frac{\mathrm{K_b K_t}}{\mathrm{R}}\dot{\alpha} \tag{1}$$

$$\mathrm{L m_p}\ddot{\alpha} + \left(\frac{\mathrm{I_w}}{\mathrm{r_w}^2} + \mathrm{m_p} + \mathrm{m_w}\right)\ddot{\mathrm{x}} = \frac{\mathrm{K_t}}{\mathrm{R r_w}}\mathrm{V} - \frac{\mathrm{K_b K_t}}{\mathrm{R r_w}^2}\dot{\mathrm{x}} + \frac{\mathrm{K_b K_t}}{\mathrm{R r_w}}\dot{\alpha}, \tag{2}$$

where the parameters are described in Subsection 2.2. Finally, the resulting state-space representation of (1) and (2) is given by

$$\dot{\xi} = \mathrm{A}\xi + \mathrm{B}u$$

$$\mathrm{y} = \mathrm{C}\xi + \mathrm{D}u$$

$$\mathrm{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{\mathrm{g q_1 q_4}}{\mathrm{q_1^2 + q_2 q_4}} & \frac{\mathrm{q_3(q_1 - q_4 r_w)}}{\mathrm{(q_1^2 + q_2 q_4) r_w}} & 0 & \frac{\mathrm{q_3(-q_1 + q_4 r_w)}}{\mathrm{(q_1^2 + q_2 q_4) r_w^2}} \\ 0 & 0 & 0 & 1 \\ \frac{\mathrm{g q_1^2}}{\mathrm{q_1^2 + q_2 q_4}} & -\frac{\mathrm{q_3(q_2 + q_1 r_w)}}{\mathrm{(q_1^2 + q_2 q_4) r_w}} & 0 & \frac{\mathrm{q_3(q_2 + q_1 r_w)}}{\mathrm{(q_1^2 + q_2 q_4) r_w^2}} \end{bmatrix}, \tag{3}$$

$$\mathrm{B} = \begin{bmatrix} 0 \\ \frac{\mathrm{k_t(q_1 - q_4 r_w)}}{\mathrm{R(q_1^2 + q_2 q_4) r_w}} \\ 0 \\ -\frac{\mathrm{k_t(q_2 + q_1 r_w)}}{\mathrm{R(q_1^2 + q_2 q_4) r_w}} \end{bmatrix}, \tag{4}$$

$$\mathrm{C} = [0 \quad 0 \quad 1 \quad 0], \mathrm{D} = [0], \tag{5}$$

where

$$\mathrm{q_1} = \mathrm{L m_p}, \qquad \mathrm{q_2} = \mathrm{I_p} + \mathrm{L^2 m_p},$$
$$\mathrm{q_3} = \frac{\mathrm{k_b k_t}}{\mathrm{R}}, \qquad \mathrm{q_4} = -\mathrm{m_p} - \mathrm{m_w} - \frac{\mathrm{I_w}}{\mathrm{r_w^2}}.$$

The model (3), (4) and (5) considers that the input (manipulated variable) is the motor voltage ($u$ = voltage). The state vector is $\xi = \begin{bmatrix} \alpha & \dot{\alpha} & \mathrm{x} & \dot{\mathrm{x}} \end{bmatrix}^{\mathrm{T}}$, where $\alpha$ is the robot's tilt angle and $\dot{\alpha}$ is its angular velocity around the center of mass; $x$ is the linear displacement of the robot and $\dot{x}$ is its linear velocity. The output (variable of interest) is the robot's linear displacement ($y = x$), while the robot's tilt angle $\alpha$ is required to remain zero (robot in balance around its vertical position).

Further details on the system modeling can be found in Howard & Bushnell (2015).

### 2.4 Prototype Instrumentation

To control the Two-Wheeled Self-Balancing Robot, all sensors and actuators must function correctly. The sensors in the prototype are an encoder, a gyroscope, and an accelerometer. The software Simulink® within the Matlab® environment and an Arduino Mega board are used for data acquisition (Minseg, 2021; Hurst, 2021).

The encoder data acquisition diagram is shown in Figure 2. The blocks responsible for acquiring the signal from both left and right motors, averaging the values, and then converting the measured values to meters are highlighted in blue. For this conversion, *pm* represents the number of pulses the encoder measures when the motor shaft rotates 360°, and this value is *pm* = 1320. The gain that converts the encoder pulses to an angle in radians is given by

$$\mathrm{E_c} = \frac{360\pi}{180\mathrm{pm}} = \frac{360\pi}{180\mathrm{x}1320} = 0.0047. \quad (6)$$

The MPU-6050 sensor is used to calculate the robot's angular position relative to its vertical position. This sensor contains both a gyroscope and an accelerometer. The signals from these sensors are processed independently, as shown in the "Gyroscope" and "Accelerometer" groups, highlighted in grey and red, respectively, in Figure 2.

Only the variable on the X-axis is used to process the gyroscope signal, representing the robot's angular velocity about its center of mass. However, a "bias" error can occur in this measure, which causes the hardware to read a value other than exactly zero when the prototype is at rest. To correct this error, a constant value can be added or subtracted from the sensor reading before integration.

The data from the axes Y and Z are obtained through the accelerometer to calculate the angular tilt of the robot. Similarly to the gyroscope, if the hardware does not read exactly zero in the vertical position, constant values can be added or subtracted in the reading to remove this error. After this correction, the signals related to these two axes pass through an arctangent block. Further details of this process can be found in Varaschim et al. (2022).

The gyroscope can measure rapid changes in rotation, but its signal, the angular velocity, can present steady-state error when integrated to obtain the angular position. The accelerometer allows us to obtain the steady-state angle accurately through the arctangent function, but it has slow dynamics in tracking motion and is susceptible to noise in measurements. One method to combine these two measurements and mitigate errors is using a complementary filter (Varaschim et al., 2022).

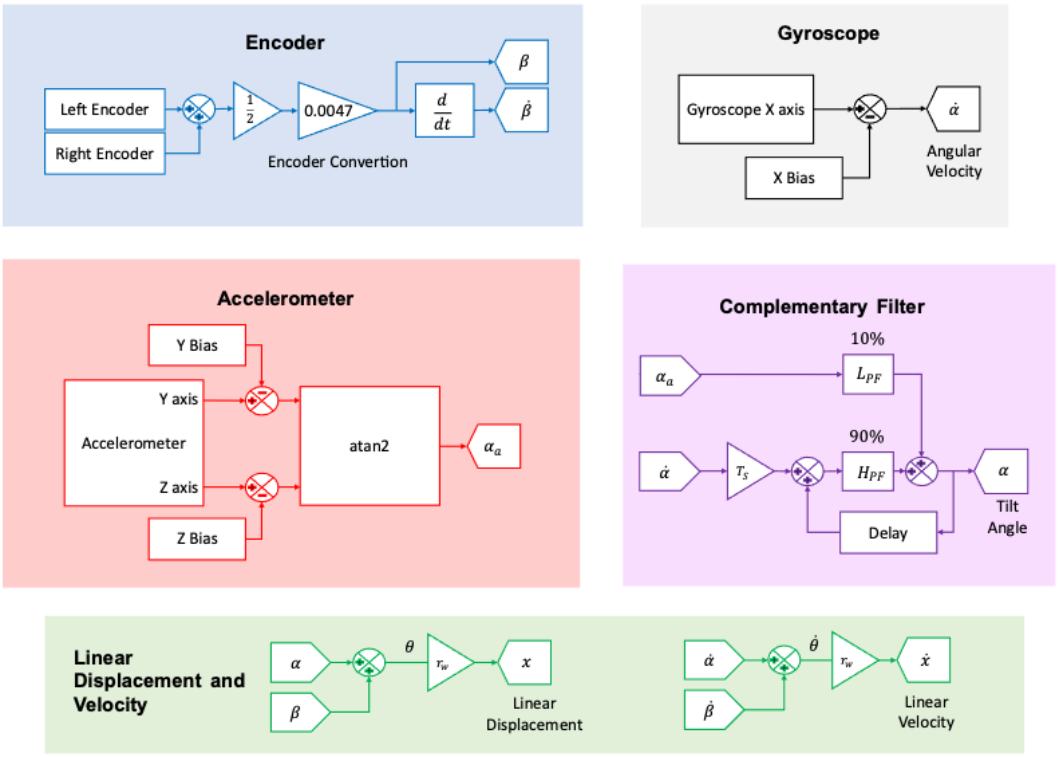


Figure 2. Signal processing for sensors in the Two-Wheeled Self-Balancing Robot prototype.

The complementary filter combines a low-pass filter for the accelerometer signal to mitigate measurement noise and a high-pass filter for the gyroscope signal to mitigate drift error. A schematic for the complementary filter applied to the MPU 6050 sensor signals is presented in Figure 3 (Mummadi et al., 2018; Varaschim et al., 2022).

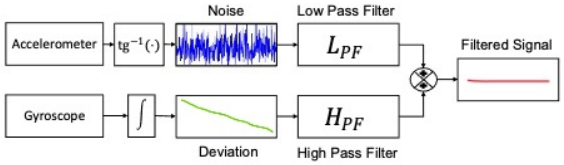


Figure 3. Complementary filter usage diagram (Mummadi et al., 2018).

The gyroscope requires only a small correction to compensate for drift accumulation, so most of the filtered angle signal comes from the gyroscope when calculating the tilt angle of the robot. In the experiments, we weighed it as 90%. However, 10% of the computed angle comes from the accelerometer to account for any accumulated drift.

This weighting between the signals of the two sensors is accomplished by a complementary filter, as shown in Figure 2 in purple (group "Complementary Filter"). The weights (10% and 90%) can be adjusted to balance response speed and steady-state error, but their sum must be 100%. The low and high pass filters, $L_{PF}$ and $H_{PF}$ respectively, that we used in the processing of the tilt angle of the robot are shown in (7) (Varaschim et al., 2022). Note that these filters are in discrete time. The sampling period used is $T_s$ = 0.02 s.

$$\mathrm{L_{PF}} = \frac{0.1\mathrm{z}}{\mathrm{z}-0.9}, \mathrm{H_{PF}} = \frac{0.9(\mathrm{z}-1)}{\mathrm{z}-0.9}. \quad (7)$$

In view of this, the state variables in (3) are measured as follows. The robot's tilt angle $\alpha$ is obtained through the complementary filter. The angular velocity of the robot $\dot{\alpha}$ is acquired through the gyroscope.

Linear displacement and velocity of the robot, represented by $x$ and $\dot{x}$, are determined by multiplying the wheel's angle $\theta$ and angular velocity $\dot{\theta}$ by the wheel's radius, respectively. Since the motors driving the wheels are located within the robot body, the angular displacement of the wheels is the sum of the angle measured by the wheel encoders, $\beta$, and the robot tilt angle, $\alpha$. This sum is due to how these angles are measured. The same logic applies to angular velocity. As illustrated in Figure 2 (in the "Linear Displacement and Velocity" group highlighted in green), $x$ and $\dot{x}$ are calculated using $\alpha$, $\beta$, $\dot{\alpha}$, and $\dot{\beta}$.

During testing, a synchronization discrepancy between the two motors that drive the robot was observed, causing the robot to deviate from a straight path. To address this issue, we developed a Proportional Integral (PI) controller to ensure one motor follows the other, thus enabling the robot to move in a straight line. Figure 4 shows a diagram of the implementation of this PI controller. In the implementation, the rotation angle of both motors is read through the encoders, which we use to calculate the rotation error between them. This error is then converted to meters (linear displacement) and used as input for the PI controller. The controller output corrects the error between the motors. Before the control signal is sent to the motors, it is converted using a polynomial. This polynomial compensates for the voltage loss between the desired voltage (sent to the motors) and the actual voltage that comes to the motor's terminal. The PI controller parameters are $K_p$ = 3 and $K_i$ = 2, determined through experimental tests with the prototype.

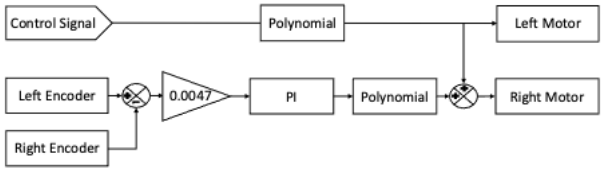


Figure 4. Drive and synchronization of robot motors.

## 2.5 Prototype Parameters' Values

We approximate the robot's body by a rigid rectangular shape of uniform mass to find its moment of inertia. This moment of inertia is calculated with respect to the motor axis (base of the robot) along the vertical length of the robot and, thus, is given by (Hibbeler, 2013)

$$\mathrm{I_p} = \frac{1}{12}\mathrm{m_p h^2} + \mathrm{m_p L^2}, \quad (8)$$

where the parameters are described in Subsection 2.2.

The armature resistance of the motor was measured using a multimeter. Utilizing the encoders on the motors, the angular velocity $\omega$ of the wheels was measured for a given voltage $V$ and current $i$. With the known resistance, voltage, current, and velocity, the electromotive force constant $K_b$ was calculated using (9). Since there are significant losses in the motor gears, it is possible to say that $K_t$ is equivalent to 65% of $K_b$ (Hurst, 2021).

$$\mathrm{V} = \mathrm{Ri} + \mathrm{K_b}\omega. \quad (9)$$

For the controller design, we considered the parameter values in Table 1, corresponding to the prototype shown in Figure 1. The parameter descriptions are given in Subsection 2.2.

Applying the values from Table 1 to (4), we find:

$$\mathrm{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 105.0769 & -11.4329 & 0 & 519.6779 \\ 0 & 0 & 0 & 1 \\ -4.2878 & 1.16 & 0 & -52.7287 \end{bmatrix}, \quad \mathrm{B} = \begin{bmatrix} 0 \\ -57.1360 \\ 0 \\ 5.7973 \end{bmatrix}. \tag{10}$$

Table 1. Two-Wheeled Self-Balancing Robot Parameters.

| Parameter | $g$ | $h$ | $K_h$ |
|---|---|---|---|
| Value | 9.81 | 0.115 | 0.2 |
| [unit] | [m/s$^2$] | [m] | [V.s/rad] |
| Parameter | $L$ | $m_p$ | $I_p$ |
| Value | 0.0575 | 0.121 | 5.36172 x 10$^{-4}$ |
| [unit] | [m] | [kg] | [kg.m$^2$] |
| Parameter | $m_w$ | $r_w$ | $I_w$ |
| Value | 0.033 | 0.022 | 7.9860 x 10$^{-6}$ |
| [unit] | [kg] | [m] | [kg.m$^2$] |
| Parameter | $R$ | $K_t$ | |
| Value | 10 | 0.13 | |
| [unit] | [Ω] | [N.m/A] | |

## 3. Controller Design

Due to hardware processing limitations, using a continuous-time control became unfeasible. Thus, we adopted the discrete-time approach.

It is possible to include an integrator with gain $K_e$ in addition to state feedback in the plant control to eliminate steady-state error. The integrator increases the system type and reduces finite error (Nise, 2010). A diagram of this control strategy, also known in the literature as integral control, tracking control, or servo system, is presented in Figure 5.

Based on the diagram in Figure 5, one finds the following dynamic equations

$$\begin{aligned} \dot{\xi} &= \mathrm{A}\xi + \mathrm{Bu} \\ \dot{\xi}_\mathrm{N} &= -\mathrm{C}\xi + \mathrm{r}, \end{aligned} \tag{11}$$

and control law

$$\begin{aligned} \mathrm{u} &= -\mathrm{K}\xi + \mathrm{K_e}\xi_\mathrm{N} = -\begin{bmatrix} \mathrm{K} & -\mathrm{K_e} \end{bmatrix} \begin{bmatrix} \xi \\ \xi_\mathrm{N} \end{bmatrix} \\ &= -\mathrm{K_N} \begin{bmatrix} \xi \\ \xi_\mathrm{N} \end{bmatrix} = -\mathrm{K_N}\hat{\xi}. \end{aligned} \tag{12}$$

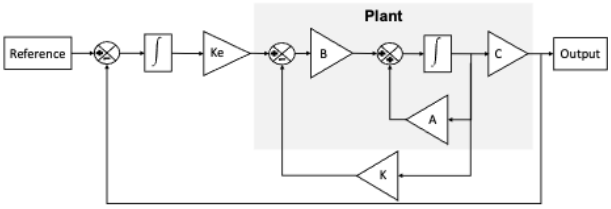


Figure 5. Servo system using state feedback (Ogata, 2009).

To design the gain matrix $K_N$ in (12), it is possible to represent the dynamics (11) using augmented matrices. Thus, the dynamics become $\dot{\hat{\xi}}(t) = A_N \hat{\xi}(t) + B_N u(t)$, where $\hat{\xi} = [\xi \ \xi_N]^T$, and the matrices are

$$\mathrm{A_N} = \begin{bmatrix} \mathrm{A} & 0 \\ -\mathrm{C} & 0 \end{bmatrix}, \text{and} \quad \mathrm{B_N} = \begin{bmatrix} \mathrm{B} \\ 0 \end{bmatrix}. \tag{13}$$

We can utilize the established design methodology for the augmented dynamics to design the gain in (12). Since we adopted the controller design in discrete time, it is necessary to discretize the dynamics. Thus, considering (13) and (10), the sampling period of $T_s$ = 0.02 s and that the control signal is applied with a Zero Order Holder (ZOH) (Franklin et al.,1998; Alves et al., 2022a), with the assistance of the Matlab® software, we obtain the discrete-time dynamics

$$\hat{\xi}(\mathrm{k}+1) = \Phi\hat{\xi}(\mathrm{k}) + \Gamma\mathrm{u}(\mathrm{k}),$$

$$\Phi = \begin{bmatrix} 1.0177 & 0.0186 & 0 & 0.0710 & 0 \\ 1.6459 & 0.8878 & 0 & 5.9019 & 0 \\ -0.0005 & 0.0002 & 1 & 0.0128 & 0 \\ -0.0387 & 0.0126 & 0 & 0.4045 & 0 \\ 0 & 0 & -0.0200 & -0.0001 & 1 \end{bmatrix}, \quad \Gamma = \begin{bmatrix} -0.0078 \\ -0.6489 \\ 0.0008 \\ 0.0655 \\ 0 \end{bmatrix}. \tag{14}$$

### 3.1 LQR control

The classical approach of LQR deals with optimizing a cost function or performance index. Thus, one can weigh which state variables and inputs as more or less important in the performance index and seek an appropriate control action to minimize it (Levine, 1996; Ogata, 2009). The discretized performance function is given by (Franklin et al., 1998; Alves et al., 2014)

$$\mathrm{J} = \sum_{\mathrm{k}=0}^{\infty} \mathrm{x(k)}^{\mathrm{T}}\mathrm{Qx(k)} + \mathrm{u(k)}^{\mathrm{T}}\mathrm{Ru}\Big(\mathrm{k}\Big). \tag{15}$$

Considering the system described by $\xi(k+1) = \phi\xi(k) + \Gamma u(k)$ and a control law $u(k) = -Kx(k)$, one can find the gain $K$ that minimizes the performance index presented in (15) by solving the discrete-time reduced Riccati matrix equation given by (Franklin et al., 1998)

$$\mathrm{S} = \Phi\Big[\mathrm{S} - \mathrm{ST}\big(\mathrm{R} + \Gamma^{\mathrm{T}}\mathrm{ST}\big)^{-1}\Gamma^{\mathrm{T}}\mathrm{S}\Big]\Phi + \mathrm{Q}. \tag{16}$$

Then, the gain $K$ is calculated as (Franklin et al., 1998):

$$\mathrm{K} = \big(\mathrm{R} + \Gamma^{\mathrm{T}}\mathrm{ST}\big)^{-1}\Gamma^{\mathrm{T}}\mathrm{S}\Phi. \tag{17}$$

The design of the gain in the LQR depends on the matrices $Q$ and $R$ in the performance index (15). We chose these matrices based on experimental tests as follows:

$$\mathrm{Q} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}, \mathrm{R} = [0.2]. \tag{18}$$

Thus, considering (14) and (18), and using the "dlqr(ϕ, Γ,Q,R)" function of the Matlab® software, we found the feedback gain $K_N$ in (12), where

$$\begin{gathered} \mathrm{K} = [\mathrm{K}_1 \quad \mathrm{K}_2 \quad \mathrm{K}_3 \quad \mathrm{K}_4] \\ = [-15.5371 \quad -2.0042 \quad -4.6868 \quad -15.5037], \\ \mathrm{K_e} = [1.6125]. \end{gathered} \tag{19}$$

### 3.2 $\mathscr{H}_\infty$ control

We design the $\mathscr{H}_\infty$ controller based on the work of Alves et al. (2022b). To do so, it is necessary to consider the model

$$\begin{gathered} \hat{\xi}(\mathrm{k}+1) = \Phi\hat{\xi}(\mathrm{k}) + \Gamma\mathrm{u(k)} + \Psi\mathrm{w(k)}, \\ \mathrm{z(k)} = \mathrm{E}\hat{\xi}(\mathrm{k}), \end{gathered} \tag{20}$$

where $\hat{\xi}(\mathrm{k}) \in \Re^{\mathrm{n_x}}$ is the state vector, $\mathrm{u(k)} \in \Re^{\mathrm{n_u}}$ is the control vector, $\mathrm{z(k)} \in \Re^{\mathrm{n_z}}$ is the system output, and $\mathrm{w(k)} \in \Re^{\mathrm{n_w}}$ is the disturbance. The matrices ϕ, Γ, Ψ and $E$ are known constant matrices with appropriate dimensions.

The design procedure consists of computationally solving the Linear Matrix Inequalities (LMIs) described in Theorem 1 and Lemma 1, simultaneously. In these LMIs, $0_{nxm}$ consists of the null matrix with $n$ lines and $m$ rolls, and $I_{\mathrm{n}}$ denotes the identity matrix of order $n$.

**Theorem 1** (adapted from Alves et al. (2020b)) Consider the system (20). For a given $1 \geq \rho > 0$, if there exist a symmetric positive definite matrix $X \in \Re^{\mathrm{n_x}\times\mathrm{n_x}}$ and a matrix $M \in \Re^{\mathrm{n_u}\times\mathrm{n_x}}$ such that

$$\begin{bmatrix} \rho^2\mathrm{X} & \Phi\mathrm{X} - \Gamma\mathrm{M} & 0_{\mathrm{n_x}\times\mathrm{n_z}} & \Psi \\ \mathrm{X}\Phi^{\mathrm{T}} - \mathrm{M}^{\mathrm{T}}\Gamma^{\mathrm{T}} & \mathrm{X} & \mathrm{XE}^{\mathrm{T}} & 0_{\mathrm{n_x}\times\mathrm{n_w}} \\ 0_{\mathrm{n_z}\times\mathrm{n_x}} & \mathrm{EX} & \mathrm{I_{n_y}} & 0_{\mathrm{n_z}\times\mathrm{n_w}} \\ \Psi^{\mathrm{T}} & 0_{\mathrm{n_w}\times\mathrm{n_x}} & 0_{\mathrm{n_d}\times\mathrm{n_z}} & \mu\mathrm{I_{n_w}} \end{bmatrix} > 0 \tag{21}$$

holds, then the control law (12), where $K_N = MX^{-1}$, assures that the closed-loop system has an $\mathscr{H}_\infty$ norm $\leqslant\mu$ and, for $w(k) = 0$, the system (12) and (20) is asymptotically stable with a decay rate greater than or equal to $\rho$.

**Proof:** see Alves et al. (2022b).

**Lemma 1** (adapted from Buzachero et al. (2012)) If there exist a symmetric positive definite matrix $X \in \Re^{\mathrm{n_x}\times\mathrm{n_x}}$, a matrix $M \in \Re^{\mathrm{n_u}\times\mathrm{n_x}}$, positive constants $\beta$ and $\mu$ such that

$$\begin{bmatrix} \mathrm{X} & \mathrm{M}^{\mathrm{T}} \\ \mathrm{M} & \beta\mathrm{I_{n_u}} \end{bmatrix} > 0, \tag{22}$$

$$\mathrm{X} > \mu_0\mathrm{I_{n_x}} \tag{23}$$

hold along with (21), then $\mathrm{K_N^T K_N} < \frac{\beta}{\mu_0^2}\mathrm{I_{n_x}}$, where $K_N = MX^{-1}$.

**Proof:** see Buzachero et al. (2012).

One can observe that model (20) includes two additional matrices (Ψ and $E$) compared to model (14), which was used for designing the LQR controller. The matrix Ψ represents how disturbances influence the system dynamics, while the matrix $E$ defines the output these disturbances affect. In this study, the disturbance is assumed to be an undesired torque associated with the robot's angular acceleration, and the objective is to minimize its impact on the robot's tilt angle. In continuous time, the Ψ matrix would be $[0 \quad 1 \quad 0 \quad 0 \quad 0]^T$, as derived from the dynamics in (3). Using the ZOH (Zero-Order Hold) method, this matrix is discretized to obtain

$$\Psi^{\mathrm{T}} = [0.0002 \quad 0.0186 \quad 0 \quad 0.0002 \quad 0]. \tag{24}$$

The output is the robot's tilt angle, leading to the matrix

$$\mathrm{E} = [1 \quad 0 \quad 0 \quad 0 \quad 0]. \tag{25}$$

LMIs (21), (22), and (23) were solved using the Matlab® software, the modeling language YALMIP (Efberg & Löfberg, 2004), and solver Sedumi (Sturm, 1999), considering the matrices in (14), (24), and (25). The parameters were set as ρ = 0.99, $\mu_0$ = 0.0002, $\beta$ = 130, $\mu$ = 15.2949. As a result, it was obtained

$$\begin{gathered} \mathrm{K} = [\mathrm{K}_1 \quad \mathrm{K}_2 \quad \mathrm{K}_3 \quad \mathrm{K}_4] \\ = [-14.8791 \; -1.7860 \; -5.6068 \; -14.5231], \\ \mathrm{K}_\mathrm{e} = [1.9577]. \end{gathered} \tag{26}$$

### 3.3 Experiments

In both experiments, we used discrete-time controllers with an integrator in the feedback loop to control the Two-Wheeled Self-Balancing Robot, enabling it to track a specified displacement reference. The control signal is calculated as in (12). Figure 6 shows how we implemented the control signal, where gains are given in (19) for the LQR controller and in (26) for the $\mathscr{H}_\infty$ controller. The state variables are $\alpha$, the tilt angle of the robot with respect to the vertical, $\dot{\alpha}$, the angular velocity of the robot with respect to its center of mass, $x$, the linear displacement of the robot, and $\dot{x}$, the linear velocity of the robot. These variables are calculated as shown in Figure 2. The control signal in Figure 6 is used to actuate the robot's motor voltage, as illustrated in Figure 4.

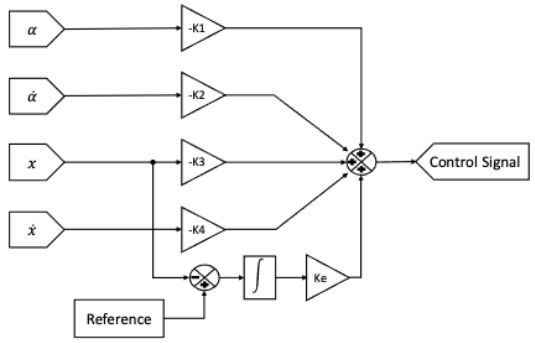


Figure 6. Control signal calculation.

## 4. Results

The discrete-time feedback gains from (19) for the LQR controller and from (26) for the $\mathscr{H}_\infty$ controller were used in simulations based on the diagram shown in Figure 5. In the experiment, we used the same gains and the diagrams shown in Figures 2, 6, and 4.

In the first set of experiments, we used a ramp-like reference for the robot's displacement. The reference is described by

$$\mathrm{r(t)} = \begin{cases} 0, 0\mathrm{s} \leq \mathrm{t} < 10\mathrm{s} \\ 0.01\mathrm{t} - 0.1, 10\mathrm{s} \leq \mathrm{t} < 40\mathrm{s} \\ 0.3, 40\mathrm{s} \leq \mathrm{t} < 70\mathrm{s} \\ -0.01\mathrm{t} + 1, 70\mathrm{s} \leq \mathrm{t} < 100\mathrm{s} \\ 0, 100\mathrm{s} \leq \mathrm{t} < 120\mathrm{s}. \end{cases} \tag{27}$$

The simulation and experimental results with reference (27) are shown in Figure 7 for the LQR controller and in Figure 8 for the $\mathscr{H}_\infty$ controller. In each figure, one can observe the linear displacement of the robot, the control signal, and the robot's tilt angle. Oscillations of the robot around the displacement reference can be noticed, with the maximum voltage not reaching 5 V and the maximum tilt angle not exceeding 0.168 rad. By the robot's tilt angle, it is clear that the robot kept its upright position.

These results were compared using performance indices (Ogata, 2009), defined as

$$\mathrm{ITAE} = \int \mathrm{t}|\mathrm{e(t)}|\mathrm{dt} \tag{28}$$

$$\mathrm{ITSE} = \int \mathrm{te(t)}^2\mathrm{dt} \tag{29}$$

$$\mathrm{IAE} = \int |\mathrm{e(t)}|\mathrm{dt} \tag{30}$$

$$\mathrm{ISE} = \int \mathrm{e(t)}^2\mathrm{dt}. \tag{31}$$

Table 2 presents the performance indices (28)-(31) calculated for the experimental tilt angle and displacement error for reference (27), where the Two-Wheeled Self-Balancing Robot was controlled using the LQR and $\mathscr{H}_\infty$ controllers. The results indicate that the $\mathscr{H}_\infty$ controller consistently achieved equal or better performance than the LQR controller, minimizing the robot's tilt angle and reducing the error between the robot's displacement and its reference trajectory.

Table 2. Performance indices (28)-(31) for real Two-Wheeled Self-Balancing Robot using ramp-like reference (27).

| Index | Tilt angle of the robot ($\alpha$) | | Linear displacement error of the robot ($x$ - $r$) | |
|---|---|---|---|---|
| | LQR | $\mathscr{H}_\infty$ | LQR | $\mathscr{H}_\infty$ |
| ITAE | 141.32 | 134.79 | 99.10 | 96.18 |
| ITSE | 5.46 | 4.84 | 2.56 | 2.32 |
| IAE | 2.51 | 2.42 | 1.74 | 1.74 |
| ISE | 0.09 | 0.08 | 0.04 | 0.04 |

In the second set of experiments, we used a sinusoidal reference given by

$$s(t) = 0.3 \sin(0.1\, t) \tag{32}$$

The simulation and the practical test results using reference (32) are presented in Figures 9 and 10 for LQR and $\mathscr{H}_\infty$ controllers, respectively. In these figures, one can observe the linear displacement of the robot, the control signal, and the tilt angle of the robot. In this experiment, there are fewer oscillations around the displacement reference compared to the previous one, with the maximum voltage not exceeding 4.5 V and the maximum tilt angle being 0.127 rad. As in the last set of tests, the robot kept its upright position.

The performance indices (28)-(31) were also employed to compare the experimental results obtained from tests conducted on the Two-Wheeled Self-Balancing Robot for the reference trajectory described in (32). Table 3 summarizes the results. It can be observed that the $\mathscr{H}_\infty$ controller consistently outperformed the LQR controller, achieving superior performance values across the evaluated metrics.

Table 3. Performance indices (28)-(31) for real Two-Wheeled Self-Balancing Robot using sinusoidal reference (32).

| Index | Tilt angle of the robot ($\alpha$) | | Linear displacement error of the robot ($x$ - $r$) | |
|---|---|---|---|---|
| | LQR | $\mathscr{H}_\infty$ | LQR | $\mathscr{H}_\infty$ |
| ITAE | 159.90 | 147.33 | 131.45 | 100.96 |
| ITSE | 5.64 | 4.72 | 3.77 | 2.23 |
| IAE | 2.84 | 2.58 | 2.34 | 1.79 |
| ISE | 0.10 | 0.08 | 0.07 | 0.04 |

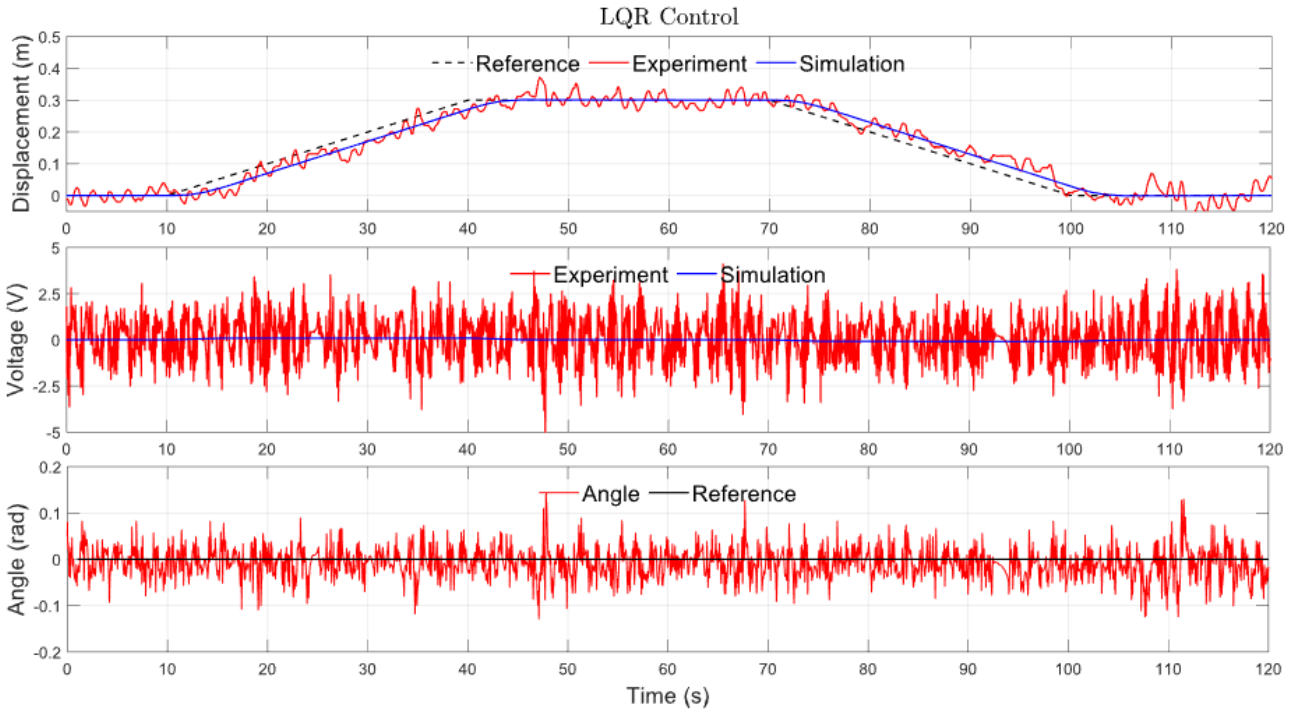


Figure 7. Linear displacement of the robot, control signal, and robot's tilt angle using reference (27) and LQR control (19).

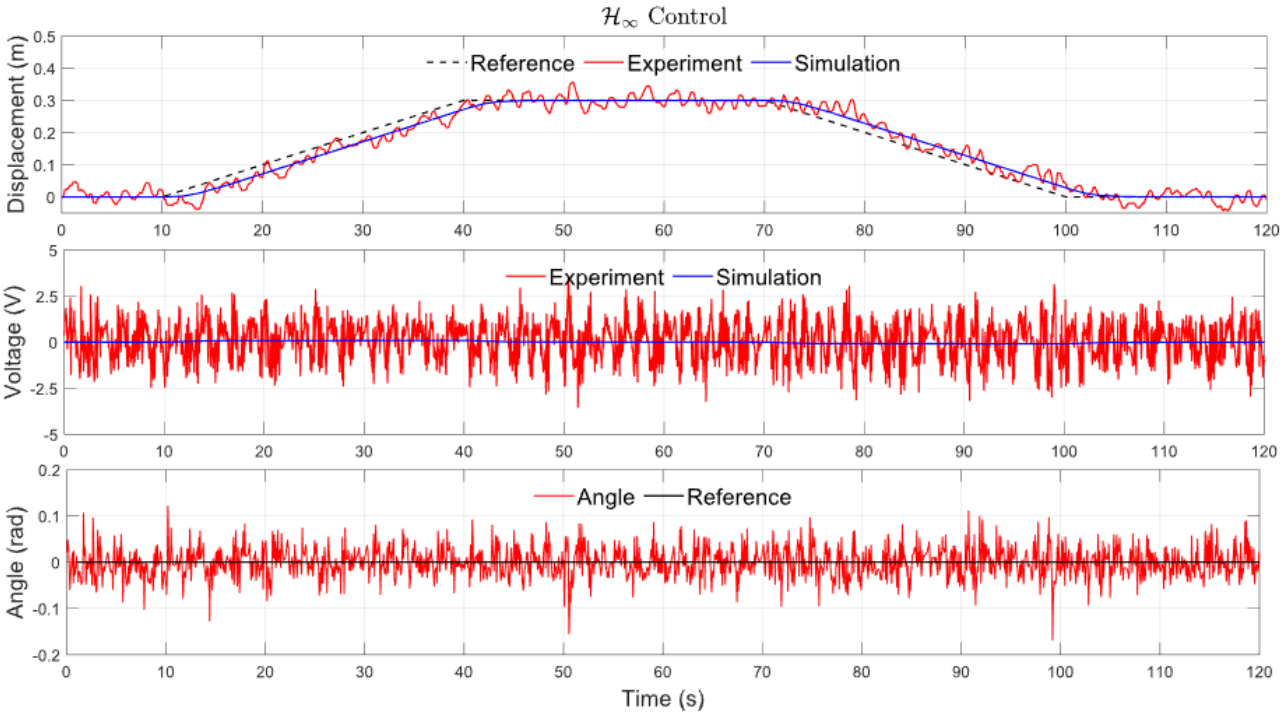


Figure 8. Linear displacement of the robot, control signal, and robot's tilt angle using reference (27) and $\mathscr{H}_\infty$ control (26).

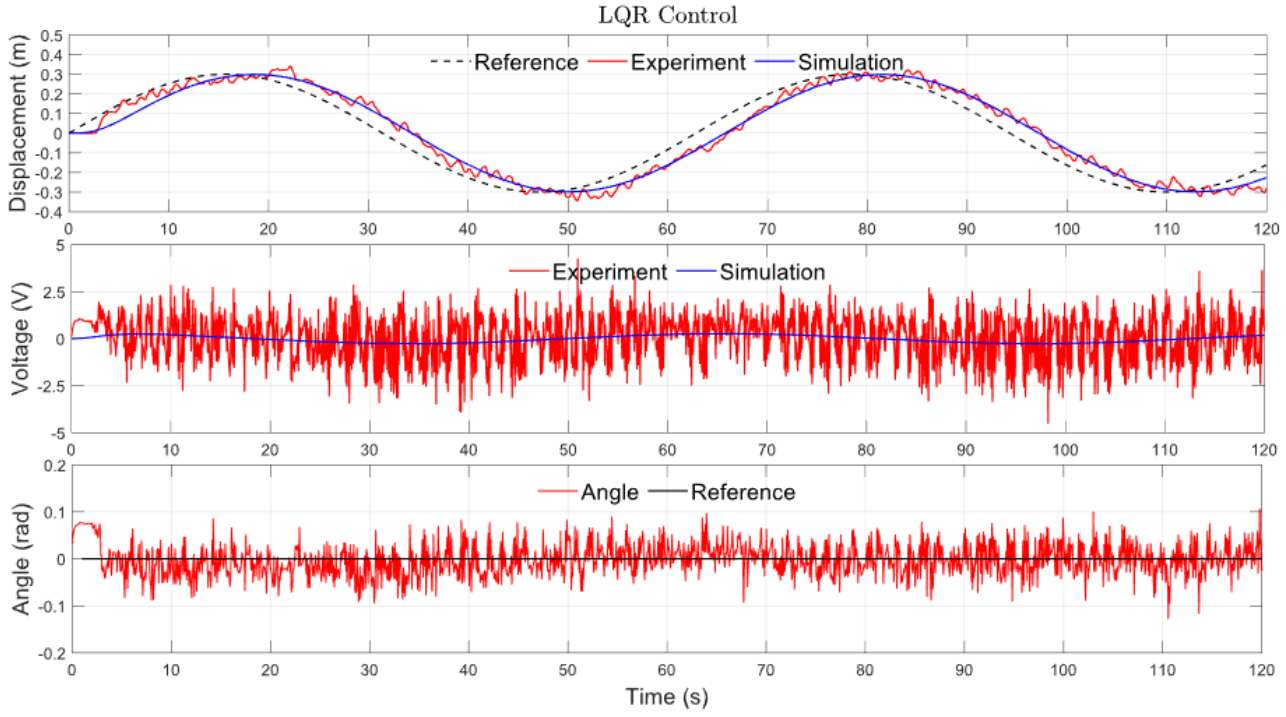


Figure 9. Linear displacement of the robot, control signal, and robot's tilt angle using reference (32) and LQR control (19).

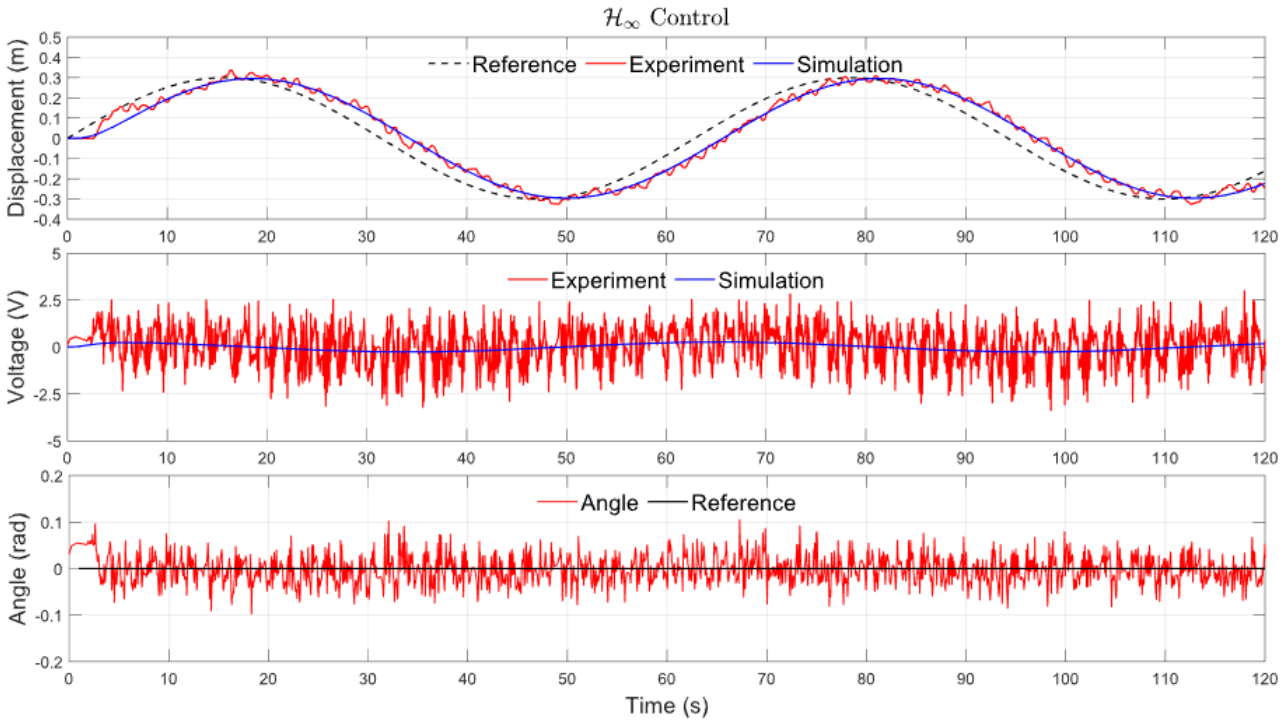


Figure 10. Linear displacement of the robot, control signal, and robot's tilt angle using reference (32) and $\mathscr{H}_\infty$ control (26).

In Figures 7-10, one can observe that the designed control strategy successfully allowed the Two-Wheeled Self-Balancing Robot prototype to follow the references for its displacement while maintaining its balance. Noises and unconsidered dynamics are present in the prototype, not the simulations. Despite that, the implemented strategies could handle this difficulty in the actual application.

The level of oscillation observed in the robot's tilt angle can be attributed to structural characteristics such as a high center of mass and narrow wheelbase. While design changes could mitigate these factors, this study focuses on developing and testing the control strategy under the current prototype configuration. Future work could explore the impact of structural optimization on control performance.

## 5. Conclusions

This paper explored the control of a Two-Wheeled Self-Balancing Robot by designing two discrete-time controllers with an integrator in the feedback control loop, employing LQR and $\mathscr{H}_\infty$ techniques. Unlike previous works in literature, the objective was to balance the robot while ensuring it followed a desired reference for its linear displacement. Simulations and experimental tests were conducted using ramp-like and sinusoidal reference signals to evaluate the robot's linear displacement performance. Both control strategies demonstrated satisfactory results, achieving the desired position while maintaining balance. During experimental tests, the $\mathscr{H}_\infty$ controller exhibited equal or superior performance compared to the LQR controller across all tested indices (ITAE, ITSE, IAE, and ISE).

While this study focused on evaluating control strategies under the current prototype configuration, it is acknowledged that specific structural characteristics, such as the robot's high center of mass and narrow base, may have contributed to observed oscillations. These design choices were intentionally maintained to test the robustness of the proposed controllers in sub-optimal conditions. Future work could explore hardware optimizations, such as lowering the center of mass or redistributing weight, to complement the presented control strategies and further enhance system stability.

Future work will still aim to mitigate displacement error further and simultaneously reduce robot oscillations concerning the vertical position by exploring alternative control techniques.

## Conflict of interest

The authors have no conflict of interest to declare.

## Acknowledgment

The authors would like to thank the Brazilian institutions, the National Council for Scientific and Technological Development (CNPq), and the Federal Institute of Paraná (IFPR), for their support in developing this research.

## Funding

This work received financial support from the National Council for Scientific and Technological Development - CNPq (123232/2023-6), Brazil.

## References

Alves, U. N. L., Breganon, R., Pivovar, L. E., de Almeida, J. P. L., Martins, L. F. B., Barbara, G. V., ... & Mendonça, M. (2022a). Discrete-time modeling and state feedback control of a DC motor with inertial load. *Journal of applied research and technology*, *20*(6), 718-728.
https://doi.org/10.22201/icat.24486736e.2022.20.6.1304

Alves, U. N. L. T., Breganon, R., Pivovar, L. E., Almeida, J. P. L. S. de, Barbara, G. V., Mendonça, M., & Palácios, R. H. C. (2022b). Discrete-time H∞ integral control via LMIs applied to a Furuta pendulum. *Journal of Control, Automation and Electrical Systems*, 33(3), 1-12.
https://doi.org/10.1007/s40313-021-00867-x

Alves, U. N. L. T., Garcia, J. P. F., Teixeira M., Garcia, S. C., & Rodrigues, F. B. (2014). Sliding mode control for active suspension system with data acquisition delay. *Mathematical Problems in Engineering*, 1-13.
https://doi.org/10.1155/2014/529293

Boyd, S., El Ghaoui, L., Feron, E., & Balakrishnan, V. (1994). *Linear matrix inequalities in system and control theory*. Society for industrial and applied mathematics.

Breganon, R., Alves, U. N. L. T., Almeida, J. P. L. S. de, Ribeiro, F. S. F., Mendonça, M., Palácios, R. H. C., & Montezuma, M. A. F. (2023). Model Identification and H∞ Control of an Aeropendulum. *Journal of Applied Research and Technology*, *21*(4), 526-534.
https://doi.org/10.22201/icat.24486736e.2023.21.4.1741

Buzachero, L. F. S., Assuncao, E., Teixeira, M. C. M., & Silva, E. R. P. da (2012). New Techniques for Optimizing the Norm of Robust Controllers of Polytopic Uncertain Linear Systems. Frontiers in Advances Control Systems, InTech, pp. 75-100.
https://doi.org/10.5772/38920

Cho, N. (2021). Nonlinear Optimal Missile Guidance for Stationary Target Interception with Pendulum Motion Perspective. *IEEE, American Control Conference (ACC).* New Orleans, USA, 3908-3913.
https://doi.org/10.23919/ACC50511.2021.9483418

Dhewa, O. A., Priyambodo, T. K., Nasuha, A., & Mustofa, Y. M. (2022). Enhancement of Stability on Autonomous Waypoint Mission of Quadrotor using LQR Integrator Control. *IIUM Engineering Journal*, *23*(1), 129-158.
https://doi.org/10.31436/iiumej.v23i1.1803

Efberg, J., & Löfberg, J. (2004). YALMIP: A Toolbox for Modeling and Optimization in Matlab. *In: Proceedings of the CACSD Conference,* 284-289. Taipei, Taiwan
https://doi.org/10.1109/CACSD.2004.1393890

Franklin, G. F., Powell, J. D., & Workman, M. L. (1998). *Digital control of dynamic systems* (Vol. 3). Menlo Park: Addison-wesley.

Graichen, K., Treuer, M., & Zeitz, M. (2007). Swing-up of the double pendulum on a cart by feedforward and feedback control with experimental validation. *Automatica*, *43*(1), 63-71.
https://doi.org/10.1016/j.automatica.2006.07.023

Hazem, Z. B., Fotuhi, M. J., & Bingül, Z. (2020). Development of a Fuzzy-LQR and Fuzzy-LQG stability control for a double link rotary inverted pendulum. *Journal of the Franklin Institute*, *357*(15), 10529-10556.
https://doi.org/10.1016/j.jfranklin.2020.08.030

Hehn, M., & D'Andrea, R. (2011). A flying inverted pendulum. In *2011 IEEE International Conference on Robotics and Automation* (pp. 763-770). IEEE.
https://doi.org/10.1109/ICRA.2011.5980244

Hibbeler, R. C. (2013). Engineering Mechanics: Statics. 13. ed. New Jersey: Pearson.
https://www.google.com.br/books/edition/Mechanics_for_Engineers/ixc5wAEACAAJ?hl=en

Howard, B., & Bushnell, L. (2015). Enhancing Linear System Theory Curriculum with an Inverted Pendulum Robot. *IEEE American Control Conference (ACC)*, 2185-2192, Chicago, USA.
https://doi.org/10.1109/ACC.2015.7171057

Hurst, J. (2021). Rensselaer Arduino Support Package Library (RASPLib), Rensselaer Polytechnic Institute, School of Engineering.

Jeong, J., Kim, S., & Suk, J. (2013). Optimal tracking control system design for a ring-wing type UAV. In *2013 International Conference on Unmanned Aircraft Systems (ICUAS)* (pp. 413-422). IEEE.
https://doi.org/10.1109/ICUAS.2013.6564716

Kim, Y., Kim, S. H., & Kwak, Y. K. (2005). Dynamic analysis of a nonholonomic two-wheeled inverted pendulum robot. *Journal of Intelligent and Robotic Systems*, *44*(1), 25-46.
https://doi.org/10.1007/s10846-005-9022-4

Levine, W. S. (1996). The Control Handbook, Boca Raton, Florida CRC Press.

Minseg, (2021). The Miniature Balancing Robot: A Low-cos Mobile Lab Experiment Kit for Education.
https://minseg.com/collections/minseg-kits/products/minsegshield-m2v5-dual-axis-balance-minseg-kit-new

Moon, H. T., Kim, H. S., & Youn, M. J. (2003). A discrete-time predictive current control for PMSM. *IEEE, Transactions on Power Electronics,* 18(1), 464-472.
https://doi.org/10.1109/TPEL.2002.807131

Mummadi, C. K., Philips Peter Leo, F., Deep Verma, K., Kasireddy, S., Scholl, P. M., Kempfle, J., & Van Laerhoven, K. (2018). Real-time and embedded detection of hand gestures with an IMU-based glove. In *Informatics* (Vol. 5, No. 2, p. 28). MDPI.
https://doi.org/10.3390/informatics5020028

Niro, L., Kaneko, E. H., Mollon, M. F., Chaves, W. S., & Montezuma, M. A. F. (2017). Control of a modified ball and beam system using tracking system in real time with a dc motor as an actuator. *International Journal of Advanced Engineering Research and Science*, 4(12), 099-107.
https://doi.org/10.22161/ijaers.4.12.17

Nise, N. S. (2010). Control Systems Engineering. 6. ed. John Wiley & Sons.
https://www.google.com.br/books/edition/Control_Systems_Engineering_Sixth_Editio/1xipuAAACAAJ?hl=pt-BR

de Oliveira, D. R., Teixeira, M. C. M., Alves, U. N. L. T., de Souza, W. A., Assunção, E., & Cardim, R. (2018). On local H∞ switched controller design for uncertain T–S fuzzy systems subject to actuator saturation with unknown membership functions. *Fuzzy Sets and Systems*, *344*, 1-26.
https://doi.org/10.1016/j.fss.2017.12.004

Ogata, K. (2009). Modern Control Engineering. 5. ed. Prentice Hall.

Park, O., Shin, H., & Tsourdos, A. (2019). Linear quadratic tracker with integrator using integral reinforcement learning. In *2019 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED UAS)* (pp. 31-36). IEEE.
https://doi.org/10.1109/REDUAS47371.2019.8999679

Pedroso, C. C. S., & Modesto, E. P. (2017). Sistema de Controle de Pêndulo Invertido. Trabalho de Conclusão de Curso em Tecnológica em Mecatrônica Industrial. Universidade Tecnologica Federal do Paraná, Curitiba-PR, Brasil. In Portuguese.

Sturm, J. F. (1999). Using Sedumi 1.02, a Matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, *11*(1-4), 625-653.
https://doi.org/10.1080/10556789908805766

Sun, L., & Gan, J. (2010). Researching of two-wheeled self-balancing robot base on LQR combined with PID. In *2010 2nd International Workshop on Intelligent Systems and Applications* (pp. 1-5). IEEE.
https://doi.org/10.1109/IWISA.2010.5473610

Tirmant, H., Baloh, M., Vermeiren, L., Guerra, T. M., & Parent, M. (2002). B2, an alternative two wheeled vehicle for an automated urban transportation system. In *Intelligent Vehicle Symposium, 2002. IEEE* (Vol. 2, pp. 594-603). IEEE.
https://doi.org/10.1109/IVS.2002.1188017

Valadão, G. B. (2022). Controle de um Robô Pêndulo Invertido sobre Duas Rodas. Trabalho de conclusão de curso de Engenharia de Computação, Universidade Tecnologica Federal do Paraná, Pato Branco-PR, Brasil. In Portuguese.

Varaschim, F. H., Possoli, F. A. A., Breganon, R., & Alves, U. N. L. T. (2023). Simulação de Controle Integral via LQR de um Robô Pêndulo Invertido. *IV SIMECA Simpósio de Engenharia de Controle e Automação*, Jacarezinho-PR, Brasil. In Portuguese.
https://doi.org/10.29327/1337590.4-4

Varaschim, F. H., Soares, J. W. M., Possoli, F. A. A., Pivovar, L. E., Breganon, R., & Alves, U. N. L. T. (2022). Utilização de um Filtro Complementar junto ao Sensor MPU 6050. *Revista Mundi Engenharia, Tecnologia e Gestão*. In Portuguese.
https://doi.org/10.21575/25254782rmetg2022vol7n72341

Yamanaka, H. F., Bispo, C. A. S., Breganon, R., Ribeiro, F. S. F., Almeida, J. P. L. S., & Alves, U. N. L. T. (2022). Construção e Controle Seguidor via LQR de um Sistema Aeropêndulo. *Anais do XXIV Congresso Brasileiro de Automática*. Fortaleza - CE, Sociedade Brasileira Automática - SBA. In Portuguese.
https://doi.org/10.20906/CBA2022/3193